PREMIS/RDF Mapping

Version: 0.7 (Alpha) 2010-03-26
Authors: Rob Sanderson (azaroth42@gmail.com)
Ed Summers,
John Harrison

General Notes

- Whenever possible, existing well known ontologies have been used rather than inventing new predicates. See table below
- The names of entries from the PREMIS data dictionary have been reduced to their semantic value, as opposed to multiplying them by their location in the structure. For example objectIdentifierValue and significantPropertiesValue both become simple pms:value, attached to an appropriate subject.
- Enumerable Types are modeled as subclasses of a more generic class, rather than being an opaque string value.
- The prefix 'pms' is for the to-be-created premis ontology
- As RDF is infinitely extensible by design, there is no need for explicit Extension blocks

Ontology Table

rdf	http://www.w3.org/1999/02/22-rdf-syntax-ns#
rdfs	http://www.w3.org/2000/01/rdf-schema#
owl	http://www.w3.org/2002/07/owl#
xsd	http://www.w3.org/2001/XMLSchema#
dc	http://purl.org/dc/elements/1.1/
dcterms	http://purl.org/dc/terms/
dcmitype	http://purl.org/dc/dcmitype/
ore	http://www.openarchives.org/ore/terms/
bibo	http://purl.org/ontology/bibo/
event	http://purl.org/NET/c4dm/event.owl#
foaf	http://xmlns.com/foaf/0.1/
doap	http://usefulinc.com/ns/doap#

Essential Properties

[Relationship] pms:value

domain: pms:Identifier, pms:StorageLocation
range: Literal

[Relationship] pms:role

domain: pms:PreservationLevelDecision, pms:PreservationLevelPolicy
range: String Literal

[Relationship] pms:rationale

domain: pms:PreservationLevelDecision
range: String Literal

[Relationship] pms:note

domain: pms:Environment, pms:Event
range: String Literal

[Relationship] pms:encoding

domain: pms:Fixity, pms:Signature
range: String Literal

[Relationship] pms:key

domain: pms:Inhibitor, pms:Signature
range: String Literal

1. Object

[Class] pms:Object

description: "The Object entity aggregates information about a digital object held by a preservation repository and describes those characteristics relevant to preservation management."

[Class] pms:Representation

subClassOf: pms:Object
subClassOf: ore:Aggregation
description: "A digital object instantiating or embodying an Intellectual Entity. A representation is the set of stored digital files and structural metadata needed to provide a complete and reasonable rendition of the Intellectual Entity."

[Class] pms:File

subClassOf: pms:Object
description: "A named and ordered sequence of bytes that is known to an operating system."

[Class] pms:Bitstream

subClassOf: pms:Object
description: "Contiguous or non-contiguous data within a file that has meaningful properties for preservation purposes. "

1.1 ObjectIdentifier

[Class, Subclassed] pms:Identifier

description: "A designation used to uniquely identify the object within the preservation repository system in which it is stored."
exampleSubClasses: DLCIdentifer

[Relationship] pms:hasIdentifier

domain: pms:Object
range: pms:Identifier

Notes:

- objectIdentifierType becomes rdf:type. objectIdentifierValue becomes pms:value.
- It is recommended that identifiers which can be represented as URIs not be represented as pms:Identifiers, but instead by referenced with owl:sameAs or ore:similarTo.
- dcterms:identifier is not used as it has a range of Literal, and there is value in maintaining the type/value distinction.

1.2 ObjectCategory

Note: Category is replaced by the subclasses of Object.

1.3 PreservationLevel

[Class] pms:PreservationLevelDecision

subClassOf: pms:Event
description: A decision as to the preservation policy to use for an object.

[Relationship] pms:hasPreservationLevelDecision

domain: pms:Object
range: pms:PreservationLevelDecision

[Class, Subclassed] pms:PreservationLevelPolicy

subClassOf: dcterms:Policy
exampleSubClasses: BitStreamPreservationPolicy, FullPreservationPolicy, ...
description: A preservation policy about the set of functions to be applied to an object.

[Relationship] pms:hasPreservationLevelPolicy

domain: pms:PreservationLevelDecision
range: pms:PreservationLevelPolicy

Notes:

- The per-object decision and the general policy that was decided upon have been split up, as they are different entites. For example, the agent responsible for the decision to apply the policy is very likely different to the agent responsible for the policy itself.
- preservationLevelValue comes from a controlled vocabulary and hence is a type, which is mapped to a set of classes.
- preservationLevelRole is mapped to pms:role, and can be applied to either the Decision or the Policy.
- preservationLevelRationale is only applied to the decision.
- preservationLevelDateAssigned should be dcterms:created on the Decision
- Also used should be dcterms:creator on both Decision and Policy

1.4 SignificantProperties

Note: Significant properties should be recorded using a regular property. For example, if page width being 210 mm is important for an object, then the RDF should include the triple: object hasPageWidth "210mm" or similar.

1.5 ObjectCharacteristics

[Class] pms:Characteristic

[Relationship] pms:hasCharacteristic

domain: pms:Object
range: pms:Characteristic

[Relationship] pms:compositionLevel

domain: pms:Characteristic
range: Integer Literal
description: "An indication of whether the object is subject to one or more processes of decoding or unbundling. A file or bitstream can be encoded with compression, encryption, etc., or bundled with other files or bitstreams into larger packages. Knowing the order in which these actions are taken is important if the original object or objects must be recovered. "

1.5.2 Fixity

[Class, Subclassed] pms:Fixity

exampleSubClasses: MD5Fixity, SHA1Fixity, SHA256Fixity

[Relationship] pms:hasFixity

domain: pms:Object, pms:Characteristic
range: pms:Fixity

Notes:

- Algorithm is an enumerable type, hence we use the subclass system
- For the digest, we use pms:value
- The originator is either the software agent responsible for the creation of the fixity, and hence we use dcterms:creator
- Encoding is also important (hex vs base64 vs integer)

1.5.3-5 Size, Format, CreatingApplication

Notes:

- size is mapped to dcterms:extent
- format is mapped to dcterms:format. All information about the format itself is offloaded to the format registry
- creatingApplication is mapped to dcterms:creator
- dateCreatedByApplication is mapped to dcterms:created
- All can be applied to either a Characteristic, or an Object

1.5.6 Inhibitors

[Class, Subclassed] pms:Inhibitor

exampleSubClasses: DESInhibitor, PGPInhibitor, PasswordInhibitor

[Relationship] pms:hasInhibitor

domain: pms:Object, pms:Characteristic
range: pms:Inhibitor

[Relationship] pms:target

domain: pms:Inhibitor
range: String Literal

1.6 OriginalName

[Relationship] pms:originalName

domain: pms:Object, pms:Characteristic
range: String Literal

1.7 Storage

[Class, Subclassed] pms:StorageLocation

exampleSubClasses: FilePathLocation, URILocation

[Relationship] pms:hasStorageLocation

domain: pms:Object
range: pms:StorageLocation

Notes:

- pms:value should be used from the pms:StorageLocation
- There are very many hardware/filesystem/software/collection structure hierarchies. Far too many to sensibly model using PREMIS, as opposed to a more specialized ontology.
- dcterms:isPartOf is suggested as a means of linking directory/collection to disk partition to physical storage device.

1.8 Environment

[Class, Subclassed] pms:Environment

exampleSubClasses: RenderEnvironment, EditEnvironment, PrintEnvironment

[Relationship] pms:hasEnvironment

domain: pms:Object
range: pms:Environment

[Relationship] pms:quality

domain: pms:Environment
range: String Literal

Notes:

- As Purpose is enumerable, it is used as a Class
- Characteristic has another definition, so it has been renamed to quality
- dcterms:requires is used for dependencies
- dcmitypes:Software is used for the class of descriptions of software
- swName is dc:title
- swVersion is doap:Version, a class that then can have a name, date etc associated
- The literal for version (eg 1.0) then goes in doap:revision
- swType becomes subclasses of dcmitype:Software
- xxx:Hardware is used for descriptions of hardware

1.9 Signature

[Class, Subclassed] pms:Signature

exampleSubClasses: DSASHA1Signature, RSASHA1Signature

[Relationship] pms:hasSignature

domain: pms:Object
range: pms:Signature

Notes:

- Encoding is used for signatureEncoding (see Fixity)
- Method is used for Class
- value is used for signatureValue
- rules is a property of the software, not the signature. Otherwise, use pms:note.
- properties should be included using other ontologies
- pms:key for keyInformation (see Inhibitor)

1.10-13 Relationship, Events

Notes:

- Relationships should be encoded using regular RDF
- Event linking should be covered by the Event system

2. Events

[Class, Subclassed] pms:Event

subClassOf: event:Event
exampleSubClasses: CaptureEvent, CompressionEvent, CreationEvent, DeaccessionEvent, DecompressionEvent, DecryptionEvent, DeletionEvent, ...

Notes:

- The event ontology defines all of the necessary parts of an event
- 2.1 As the identifier is system generated, it should be a URI and hence does not need a pms:Identifier.
- 2.2 The eventType is subsumed into the subclasses of pms:Event
- 2.3 event:time is used for eventDateTime
- 2.4 pms:note is used for eventDetail
- 2.5.2 pms:note is used for eventOutcomeDetailNote

3. Agent

Notes:

- dcmitype:Agent is used for the class
- 3.1 If legacy identifiers are required, use pms:Identifier
- 3.2 agentName should be foaf:name
- 3.3 agentType is subsumed into subclasses